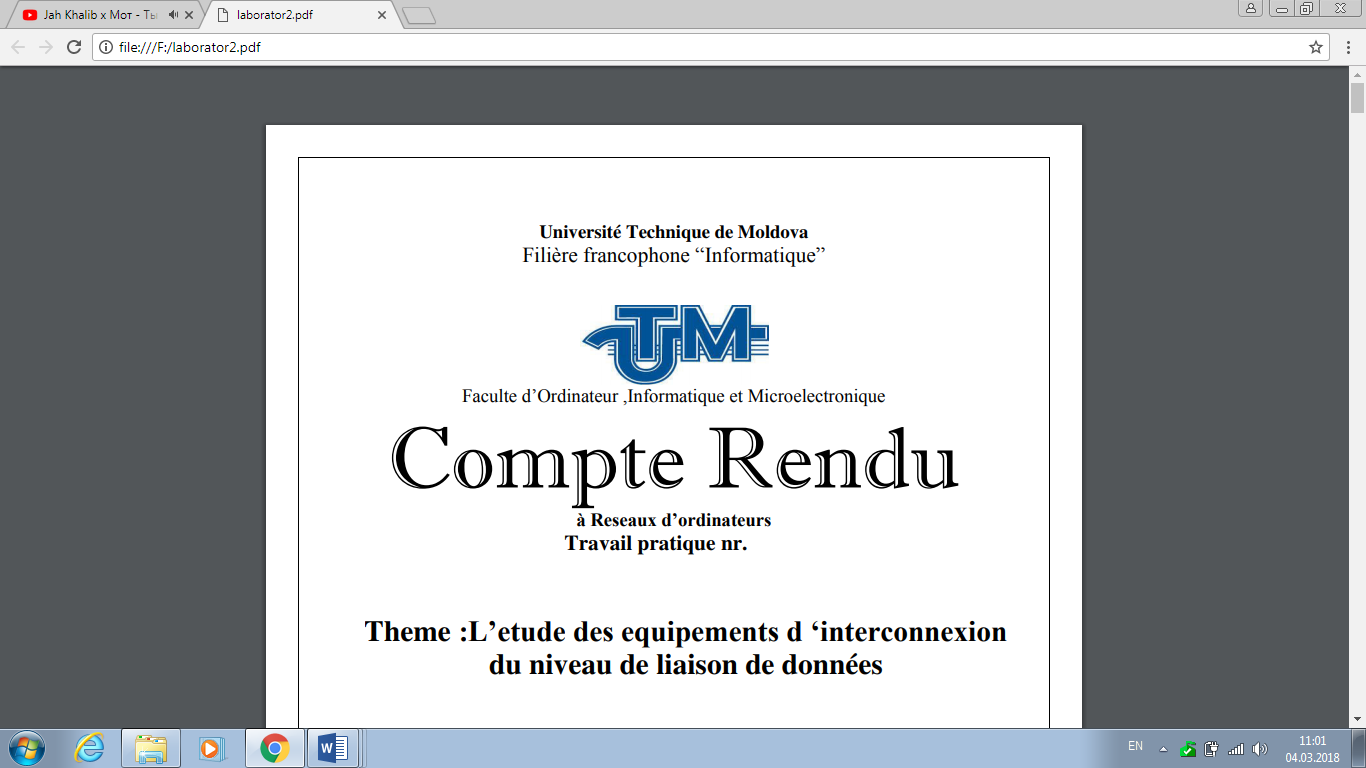
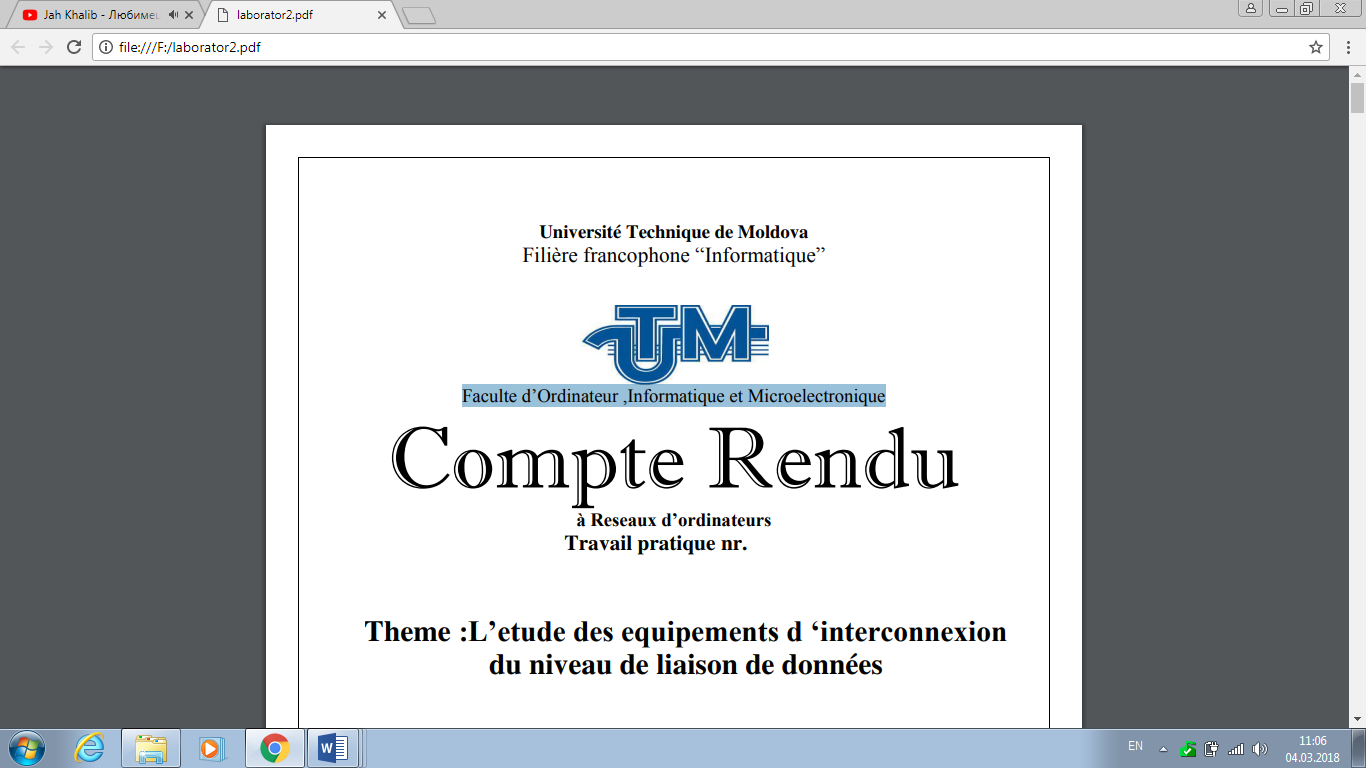
Ministère de l'éducation de la République de Moldova

Université technique de Moldavie

Faculté d’Ordinateur, Informatique et Microélectronique

Filière francophone “Informatique”





**TIDPP**

Travail pratique nr.3

**Thème : Unit testing, Dependency Injection, Mockups**

Effectué par l’étudiant(e) de gr FI-181 : Damean Alexandra

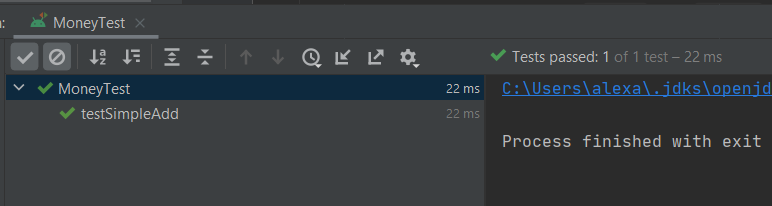
Vérifié par le professeur : Rusu Viorel

Chișinau 2020

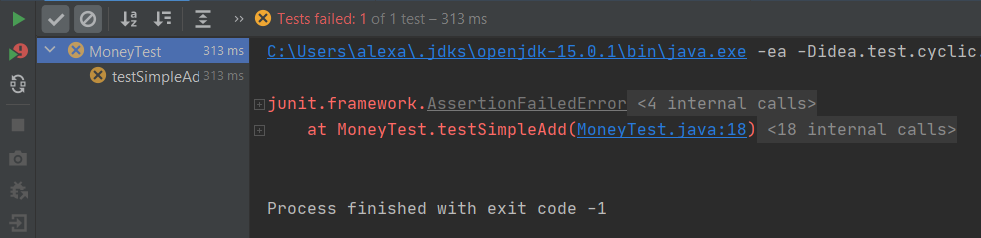
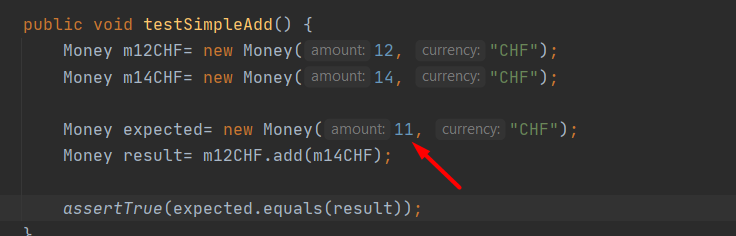
***But:*** passer par les étapes de test et de déploiement du cycle de développement d'un programme et l'utilisation d'outils Java spécifiques à ces étapes.

1. **(2.5p) Familiarisation avec l'outil JUnit.**

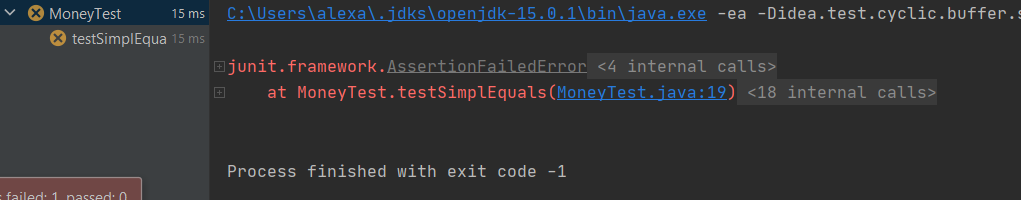
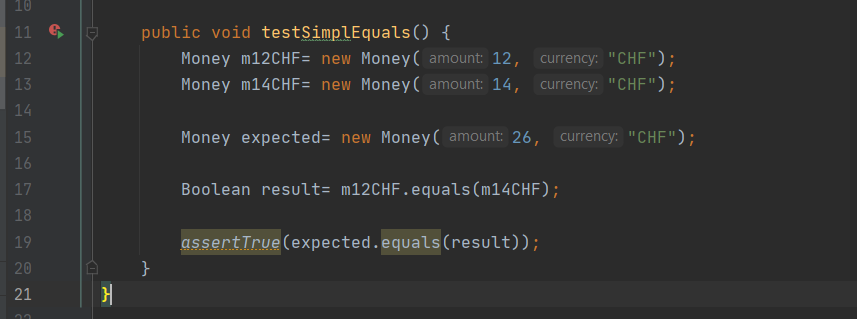
**a)**



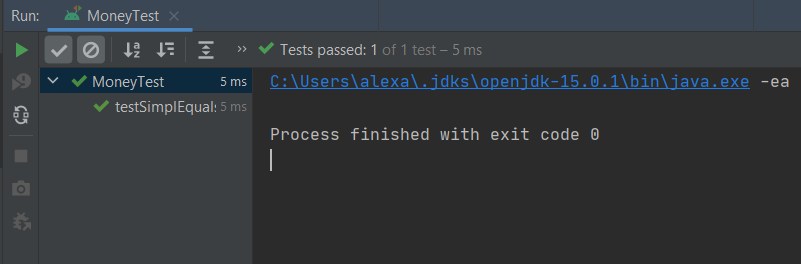
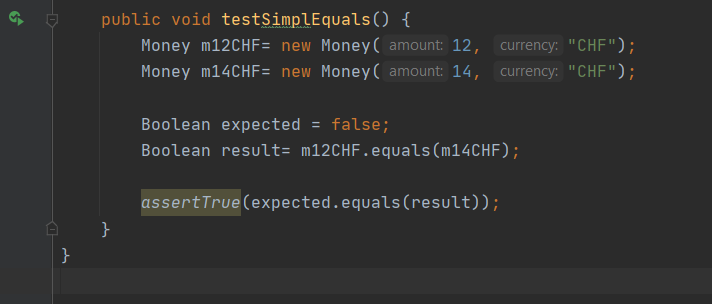
**b)**



**c)** ***mettre à jour la classe MoneyTest pour qu'elle appeler la méthode assertEquals sur les 2 objets***



***Apportez les modifications nécessaires dans la classe Money afin que le résultat soit celui souhaité.***



**2. (5p) Test de la boîte noire. Cet exercice vise à identifier les cas de test, strictement basés sur la spécification, dans manque d'accès au code source et connaissance du mode de fonctionnement interne du système.**

**Class Triangle:**

package Triangles;//  
// Source code recreated from a .class file by IntelliJ IDEA  
// (powered by Fernflower decompiler)  
//  
  
public class Triangle {  
 int a;  
 int b;  
 int c;  
  
 public Triangle(int var1, int var2, int var3) throws InvalidTriangleException {  
 this.a = var1;  
 this.b = var2;  
 this.c = var3;  
 int var4 = (var1 + var2 + var3) / 2;  
 boolean var5 = var1 == 0 || var2 == 0 || var3 == 0 || var1 < 0 || var2 < 0 || var3 < 0;  
 if (var5) {  
 throw new InvalidTriangleException();  
 }  
 }  
  
 public boolean isScalene() {  
 return this.a != this.b && this.a != this.c && this.b != this.c;  
 }  
  
 public boolean isIsosceles() {  
 return this.a == this.b || this.a == this.c || this.b == this.c;  
 }  
  
 public boolean isEquilateral() {  
 return this.a == this.b && this.b == this.c;  
 }  
}

**Class TriangleTest:**

package Triangles;  
  
import junit.framework.TestCase;  
  
public class TriangleTest extends TestCase {  
  
 public TriangleTest(String name) {  
 super(name);  
 }  
  
 // On verifie le constructeur  
  
 public void testConstructor1() throws InvalidTriangleException {  
 Triangle tr = new Triangle(5, 1, 2);  
 }  
 public void testConstructor2() throws InvalidTriangleException {  
 Triangle tr = new Triangle(1, 5, 2);  
 }  
 public void testConstructor3() throws InvalidTriangleException {  
 Triangle tr = new Triangle(2, 5, 1);  
 }  
 public void testConstructor4() throws InvalidTriangleException {  
 Triangle tr = new Triangle(5, 0, 2);  
 }  
 public void testConstructor5() throws InvalidTriangleException {  
 Triangle tr = new Triangle(0, 5, 2);  
 }  
 public void testConstructor6() throws InvalidTriangleException {  
 Triangle tr = new Triangle(2, 5, 0);  
 }  
 public void testConstructor7() throws InvalidTriangleException {  
 Triangle tr = new Triangle(0, 0, 0);  
 }  
 public void testConstructor8() throws InvalidTriangleException {  
 Triangle tr = new Triangle(5, -5, 2);  
 }  
 public void testConstructor9() throws InvalidTriangleException {  
 Triangle tr = new Triangle(-5, 5, 2);  
 }  
 public void testConstructor10() throws InvalidTriangleException {  
 Triangle tr = new Triangle(2, 5, 3);  
 }  
 public void testConstructor11() throws InvalidTriangleException {  
 Triangle tr = new Triangle(-4,-5,-3);  
 }  
  
 // On verifie la methode isScalene  
  
 public void testIsScalene1() throws InvalidTriangleException {  
 Boolean expected = false;  
 Triangle tr = new Triangle(3,4,3);  
 Boolean ourTr = tr.isScalene();  
 *assertEquals*(expected,ourTr);  
 }  
  
 public void testIsScalene2() throws InvalidTriangleException {  
 Boolean expected = false;  
 Triangle tr = new Triangle(3,3,3);  
 Boolean ourTr = tr.isScalene();  
 *assertEquals*(expected,ourTr);  
 }  
  
 // On verifie la methode isIsosceles  
  
 public void testIsIsosceles1() throws InvalidTriangleException {  
 Boolean expected = false;  
 Triangle tr = new Triangle(3,3,3);  
 Boolean ourTr = tr.isIsosceles();  
 *assertEquals*(expected,ourTr);  
 }  
 public void testIsIsosceles2() throws InvalidTriangleException {  
 Boolean expected = false;  
 Triangle tr = new Triangle(5,4,3);  
 Boolean ourTr = tr.isIsosceles();  
 *assertEquals*(expected,ourTr);  
 }  
  
 // On verifie la methode isEquilateral  
  
 public void testIsEquilateral1() throws InvalidTriangleException {  
 Boolean expected = false;  
 Triangle tr = new Triangle(5,4,3);  
 Boolean ourTr = tr.isEquilateral();  
 *assertEquals*(expected,ourTr);  
 }  
 public void testIsEquilateral2() throws InvalidTriangleException {  
 Boolean expected = false;  
 Triangle tr = new Triangle(5,4,5);  
 Boolean ourTr = tr.isEquilateral();  
 *assertEquals*(expected,ourTr);  
 }  
  
}